SORD

HANDY BUSINESS COMPUTER

M20/23 Series

OS

OPERATING SYSTEM

SORD COMPUTER SYSTEMS,INC.

PREFACE

Microcomputers have been greatly modified and enhanced since their introduction a few years ago. Even though there is a widely-accepted ratio of 4:1 hardware to software costs for large-scale systems, no such established figures exist no yet for the rapidly changing microcomputer field.

Although costs are certainly a factor in introducing any system, the key concern with microcomputers has always been their relative ease of operation. Thus, the main thrust in microcomputer design has been to enhance system operation.

This manual describes the M20/23 Series Operating System (OS), an extended and enhanced microcomputer system which uses floppy disks the key features of the M20/23 Series OS are:

(1) Beginners can use it easily.

(2) Anyone with mini-computer experience can use the same procedures to operate the M20/23 OS.

(3) File handling is very simple.
Peripheral devices can be handled using file names.

(4) Data and program file editing are both very simple.

(5) Assembler language gives you access to most of the OS functions.

(6) BASIC, Fortran-IV, and Assembler system languages and utility programs can all be used under OS control.

(7) When increasing external storage, all programs under OS control are easy to replace.

There are many more functions available in addition to these.

This manual itself supplies all the information you need to generate and handle programs for the M20/23 Series OS.

October, 1981

CONTENTS

PREFACE

1. OPERATING SYSTEM 1

1.1 Roles and Functions of the Operating System 1

1.1.1 The Role of M20/23 OS 1

1.1.2 Operating System Functions 2

1.2 Outline of the Operating System of the M20/23 Series Computers ... 3

1.2.1 Loading the Operation System 3

1.2.2 The M20/23 Series Computer Operating System 5

1.3 Bootstrap Function of the Operating System 7

1.4 Outline of the M20/23 Mark III Operating System 9

1.4.1 Loading the Operating System 9

1.4.2 M20/23 Mark III Memory Map 10

1.4.3 M20/23 Mark III System Area 10

1.6 Files 11

1.6.1 Characteristics of File Handling 11

1.6.2 Names and Functions of I/O Files 11

1.7 Disk Files 12

1.7.1 Data Files 12

1.7.2 Locations of Data on Floppy Disks 13

1.7.3 Naming Floppy Disk Data Files 14

1.7.4 File Names for Multiple Numbers of Disk Drives 14

1.7.5 File Name Extension 15

1.7.6 Organization of Disk Files 16

1.7.7 Disk 18

1.7.8 Floppy Disk for M20/23 20

1.8 Data Transfer between Main Memory and External Devices via Channel 23

1.9 Sector Sequence 26

2. OPERATING SYSTEM 27

2.1 Initializing Disks and Copying Disks 30

2.1.1 Disk Initialization 30

2.1.2 Copying Disks 31

2.1.3 Data Files 32

2.1.4 Extended COPY Functions 33

2.2 Setup 35

2.3 Setting the Date 38

2.4 Checking Unused Areas and Contents of Disks 39

2.4.1 Checking the Used and Unused Areas 39

2.4.2 Listing File Names 39

2.4.3 Extended Switches for the LIST Command 44

2.5 Creating Disk Files 46

2.5.1 Sweeping Disk Files 46

2.5.2 File Creation 47

2.6 Closing Data Files 52

2.7 Renaming Files and Changing File Characteristics 55

2.7.1 Renaming Files 55

2.7.2 Changing File Characteristics 56

2.8 Deleting and Transferring Files 58

2.8.1 Deleting Files 58

2.8.2 Transferring Files 59

2.9 Loading a New Revised Version of the Operating System 64

2.10 Programmed Function Keys 67

2.10.1 Overview 67

2.10.2 Using Programmed Function Keys 67

2.10.3 Defining Programmed Function Keys 67

2.10.4 Turn-Key Function 71

2.11 Data Medium Protection 72

2.12 Execution of BASIC Program 74

2.13 CLI Compound Commands 76

2.14 OS Error Messages 77

2.15 Disk Initialization Error 80

2.16 CLI Command Summary 82

1. OPERATING SYSTEM

The operating system is simply a system of programs which is the basis for operating the computer. In the early days of computer development there were some machines which had no operating systems, and this included even medium-sized computers. Today, even many small computers are equipped with operating systems. The more recent operating systems are so complex that it can be quite difficult to distinguish the functions of the operating system from those of the processor. A thorough understanding of the operating system is essential if effective use is to be made of today's computers. This does not mean, however, that today's computers are more difficult to use. On the contrary, they are more adaptable and responsive to user's needs. This is what the M20/23 OS is all about.

1.1 Roles and Functions of the Operating System

1.1.1 The Role of M20/23 OS

The operating system is the interface which connects the operator to the computer hardware. It consists of programs either written by or supplied to the user and it enables the hardware to be used effectively. Figure 1.1-1 shows the relationships between computer hardware, OS, and the user.

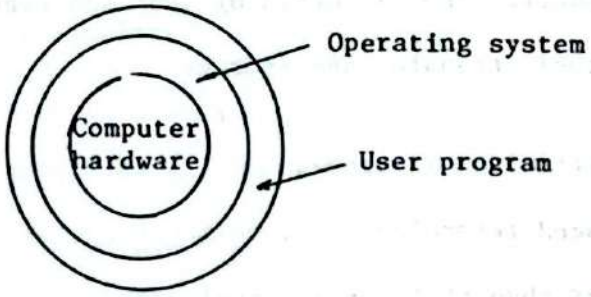


Fig. 1.1-1 Operating System Relationships

If there is no OS supplied with the computer system, the user must write a program or programs whose functions replicate those of an OS. This can, very simply, be an enormous job.

The OS makes system operation much easier, easy enough even for beginners, and enables users to take full advantage of a computer's functions with very little effort.

The goals of a good OS are:

1) To make operation of the computer system as efficient as possible,
2) To make the I/O device management more effective and provide those system functions which are necessary for simple object program generation, and
3) To make general operation of the computer much easier.

As you will find out later, OS consists of many complex programs. The user, however, need not be concerned with any program's complexity. The only thing that is important is to know what these programs can do.

1.1.2 Operating System Functions

OS is designed to:

1) Manage I/O devices
2) Manage data files
3) Control the execution of jobs and regulate I/O signals and their transfer and storage

The efficient management of I/O devices requires complicated and advanced technology. Since the response time of peripheral devices is longer than that for internal processing, much of this technology goes into making effective use of various peripheral devices.

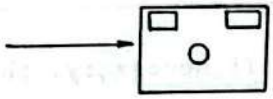
Making a programmer who is well-versed in data file management techniques before writing a program takes many years of training and on-the-job experience. The OS takes over this formidable job from the programmer so that nearly anyone can make effective use of the computer quite quickly.

1.2 Outline of the Operating System of the M20/23 Series Computers

1.2.1 Loading the Operation System

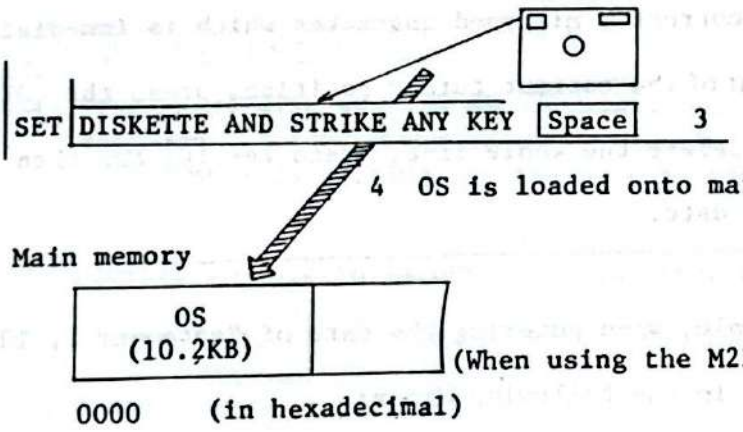
The OS software consists of various blocks.

1) The OS is stored on a floppy disk and supplied to the user as a program along with all M20/23 Series computer systems.



2) The OS is loaded into a system as shown below:

1 Turn on the power 2 Mount a disk into floppy disk drive #0.


SET DISKETTE AND STRIKE ANY KEY Space 3
4 OS is loaded onto main memory.
Main memory
OS
(10.2KB)
(When using the M23 Mark III)
0000 (in hexadecimal)

The essential part of OS is stored in a 10.2K byte resident area starting at the beginning of main memory.

3) When a program stored in OS has been read and its essential part transferred to the resident area, the following message appears on the screen:

```
     M20/M23 Mark -III DISKETTE FDDOS REV. 23D
DATE DD-MM-YY
```

This message tells the user that the M20/23 Series OS has been completely loaded.

Rev. 23D	REV. 23D at the end of the first line indicates that the OS stored in main memory is type D of the 23rd version of the operating system for the SORD M20/23 Series. When a new version is introduced, all users will be notified.

4) If necessary, the user can write the loading date on the keyboard as follows:

1. Type the date in the order and format indicated on the screen. The day, month, and year are all two digit figures are separated by hyphens.
2. To correct a mistyped character which is immediately in front of the current cursor position, press the [DEL] key.
3. To delete the whole line, press key [C] and then retype the date.

For example, when entering the date of September 1, 1981, press the keys in the following order:

[0] [1] [-] [0] [9] [-] [8] [1] [RETURN]

The date will then be printed on the screen as shown below:

```
     M20/M23 Mark -II DISKETTE FDDOS REV. 23D
 DATE DD-MM-YY 01 - 09 - 81
READY
```

If you have pressed the wrong key, press key [←] to move the cursor to the incorrect letter, press the [DEL] key to delete the letter, and then press the right key. When deleting everything, press the [C] or [CAN] key. Everything that has just been typed in will be cancelled and the system will be ready to receive new input. If it is not necessary to type the date, just press the carriage return key, [RETURN]. The ready sign and cursor will then appear on the screen and the system will be ready to receive the next instruction.

Any OS command can be executed by typing in the command and then pressing the [RETURN] key.

```
READY
  (Type the command word.)
        Press the [RETURN] key.
  [The command is executed.]
READY ← The ready sign appears on the screen to indicate that
        the system is waiting for the next instruction.
```

If the command is entered incorrectly, an error message will appear on the screen. See the section on error messages for details on what the messages mean.

1.2.2 The M20/23 Series Computer Operating System

The Operating System is stored in a floppy disk and consists of two parts:

1) The resident part

This is transferred to the residence area when the Operating System is first loaded.

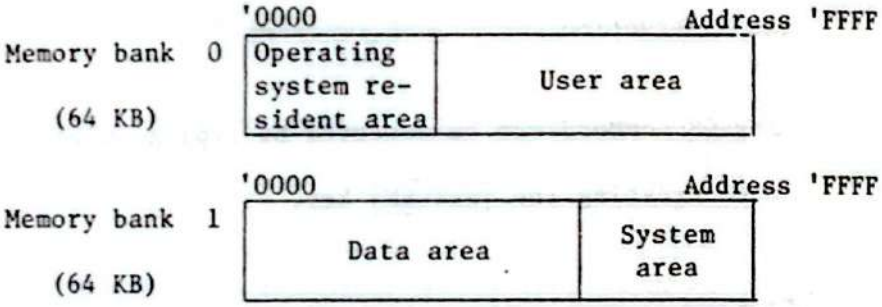
Floppy disk drive #0

2) The non-resident part

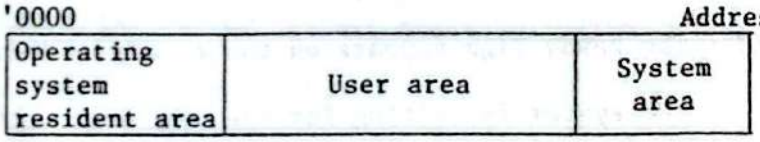
This includes any program that can be read into main memory and activated by the contents of the resident area.

3) Relationship between main memory and the operating system

① When the size of main memory (RAM) is 128K bytes, the main memory of the M23 mark III system and the operating system are:

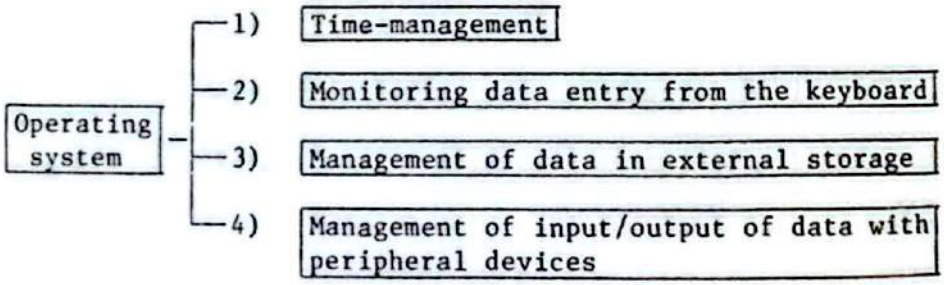


② When the size of main memory (RAM) is 64 KB bytes, the main memory of the M23 B mark III system and the operating system are:



and the system provides the resident area.

The functions of the operating system may be roughly classified into the four groups shown below.


Operating system
1) Time-management
2) Monitoring data entry from the keyboard
3) Management of data in external storage
4) Management of input/output of data with peripheral devices

Each function handles the tasks outlined below.

1) Time-management

The operating system manages the time required for inputting through the keyboard or from floppy disks.

2) Monitoring data entry from keyboard

When a program is executed, the user may wish to intervene by pressing the stop key and giving the system new instructions. The operating system monitors this entire sequence.

3) Management of data in external storage

The operating system supervises every area of the floppy disk so that the user can transfer data between the system and the floppy disk simply by entering the name of the file to be transferred. This name is called a file name.

4) Management of data input and output with peripheral devices

The operating system takes care of transfer of data between M20/23 Series machines and peripheral devices such as line printers. The user need only enter the correct instruction.

1.3 Bootstrap Function of the Operating System

The M20/23 Series operating system is stored on a floppy disk and is not loaded into main memory when the power is turned on. Actual loading is handled by a bootstrap program (ROM) which is included with the system. The bootstrap program operates as shown below.

1) Power is turned on.

When the M23 mark III system is used, power for the main frame, display, and floppy disk must all be turned on.

2) The following display appears on the screen when the power switch is turned on:

```
SET DISKETTE AND STRIKE ANY KEY-
```

You are now ready to mount a floppy disk onto drive # 0.

3) After mounting a floppy disk onto drive #0, press the space key.

This activates the bootstrap program.

The bootstrap program transfers the operating system from the floppy disk into main memory. This operation is called bootstrapping.

4) The system reads the bootstrap program during bootstrapping; the red lamp will be on.

5) When the following message appears on the screen, the bootstrap program has been executed normally:

```
M20/M23 Mark - III DISKETTE FDDOS REV. 23D
DATE DD-MM-YY
```

6) Invalid bootstrapping

Errors may occur during bootstrapping.

The system informs the user of any errors by sounding the buzzer or turning on a lamp.

The following table lists the errors indicated by the lamps:

Lamp				Error cause
SMALL	KANA	S1	S2	
o				Bootstrapping check-sum error
	o			Hardware error: invalid operation of LSI in a main frame or a bank change instruction.
		o		Memory error
o	o			Error on a floppy disk or other media

o Indicates which lamps light.

The [SMALL] key is located to the left of the space key.

The [KANA] key is located to the right of the space key.

S1 and S2 are sense switches located at the top of the keyboard.

1.4 Outline of the M20/23 Mark III Operating System

1.4.1 Loading the Operating System

1) Switch power key ON.

   At this point, there must be no floppy disk mounted on drive #0.

   The following message appears on the scrren:

```
SET DISKETTE AND STRIKE ANY KEY
```

   At this point, mount the floppy disk containing the operating system onto drive #0 and press the [space] key.

2) The lamp on drive #0 will light and the operating system will be loadded into main memory.

   The following message appears on the screen when loading is complete:

```
    M20/M23 Mark - III DISKETTE FDDOS REV. 23D
DATE DD-MM-YY
```

3) When it is, for example, September 1, 1981, the date is entered as follows:

   [0] [1] [-] [9] [-] [8] [1] now press the [RETURN] key.

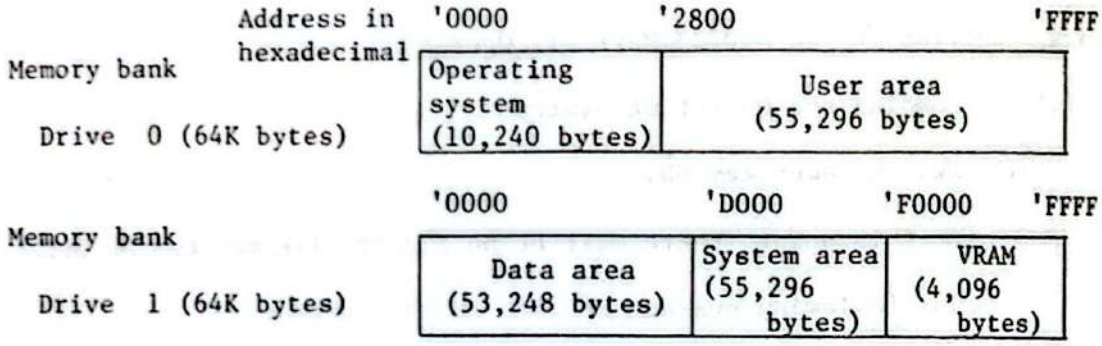
```
       M20/M23 Mark - III 1 DISKETTE FDDOS REV. 23D
  DATE DD-MM-YY 1 - 9 - 81
READY
-
```

   After you enter the date, the system returns to read status.

   The system receives the data entry with the CLI option.

1.4.2 M20/23 Mark III Memory Map

M20/23 Mark III contains 128K bytes of RAM. The memory map controls the operation of each user program including the operation system. The M20/23 Mark III memory map is diagrammed below.



1.4.3 M20/23 Mark III System Area

The M20/23 Mark III system area is located at addresses between '0000 and 'F000 of memory bank 1. This area is not used for actual operation. The following table lists the addresses of the system area:

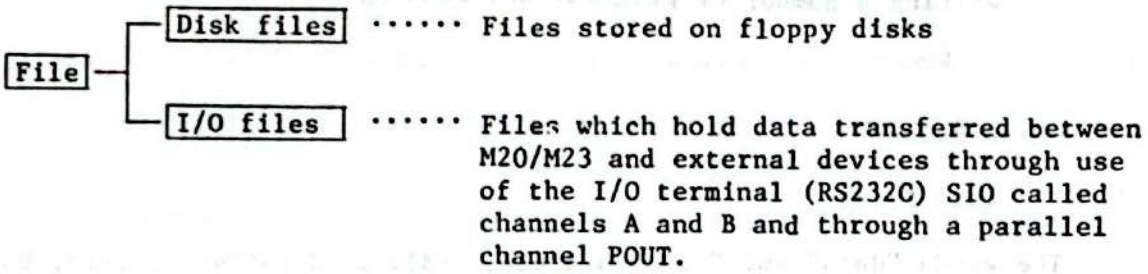
Memory bank 1 addresses in hexadecimal	Function
'D000 ∿ 'DDFF	Reserved area
'DE00 ∿ 'E1FF	Error message
'E200 ∿ 'E2FF	Printer conversion table
'E300 ∿ 'E3FF	Reserved area
'E400 ∿ 'E5FF	Keyboard conversion table
'E600 ∿ 'E7FF	Function key table
'E800 ∿ 'EFFF	Channel buffer area (Eight channels of 256-byte channel buffers)

1.6 Files

A file is a collection of data. Files of data are read and written depending on program instructions. For the M20/M23 Series, files may be considered to be collections of recorded memory on floppy disks.

In this sense a file of data is simply called a file.

There are two types of files used in the M20/M23 Series:


File
Disk files
Files stored on floppy disks
I/O files
Files which hold data transferred between M20/M23 and external devices through use of the I/O terminal (RS232C) SIO called channels A and B and through a parallel channel POUT.

1.6.1 Characteristics of File Handling

The M20/M23 Series operating system can handle data files and I/O files only when they have names. When writing a program, the user just writes the names of the files to be used. The operating system will then be able to process them.

1.6.2 Names and Functions of I/O Files

The I/O file consists of SIO channels A and B and POUT. The operating system names this file itself.

1) SIO (Serial input and output)

Channel A		Channel B	
Name	Function	Name	Function
SINA	Input of data through Channel A	SINB	Input of data through Channel B
SOUTA	Output of data through Channel A	SOUTB	Output of data through Cnannel B

2) POUT (Parallel)

POUT is the name of a file used when data is output through a parallel channel. Data entry refers to entering data from a peripheral device to a computer system.

Response data is then output.

You will acquire a really good understanding of this only by writing a number of programs and running them on M20/M23.

1.7 Disk Files

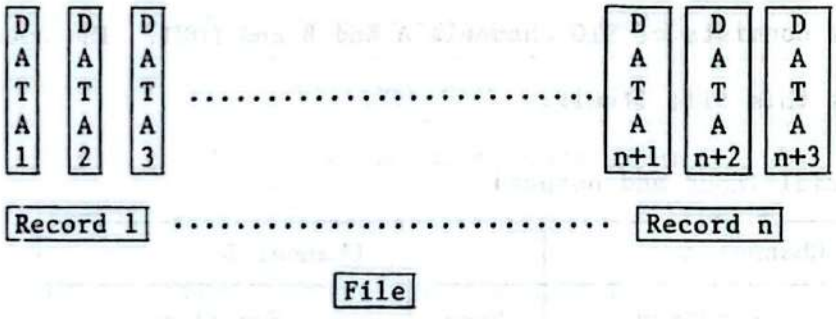
1.7.1 Data Files

The words "data" and "file" are admittedly used rather vaguely, but you should now have a general idea of their meanings. In this section, instead of giving precise definitions as is customarily done in mathematics test books, these words' meanings and inter-relationships will be explained in a general way.

In this manual: "Data" means information on an object

"Record" means a collection of data

"File" means a collection of records

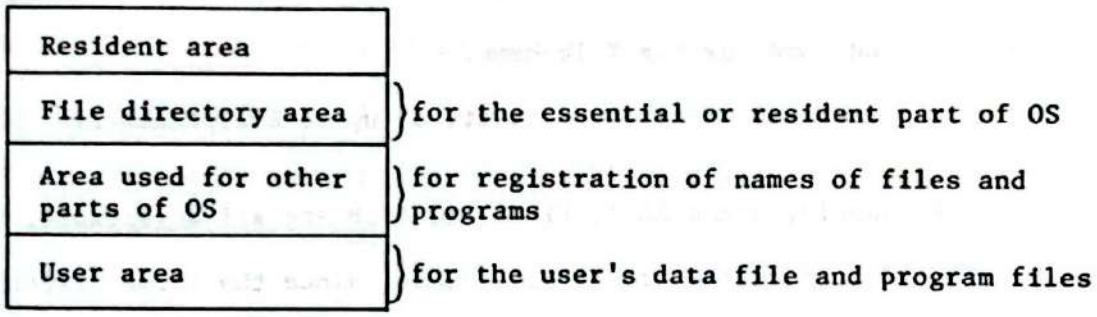


Even though some computer-makers have definitions somewhat different from those above, these are the most common.

1.7.2 Locations of Data on Floppy Disks

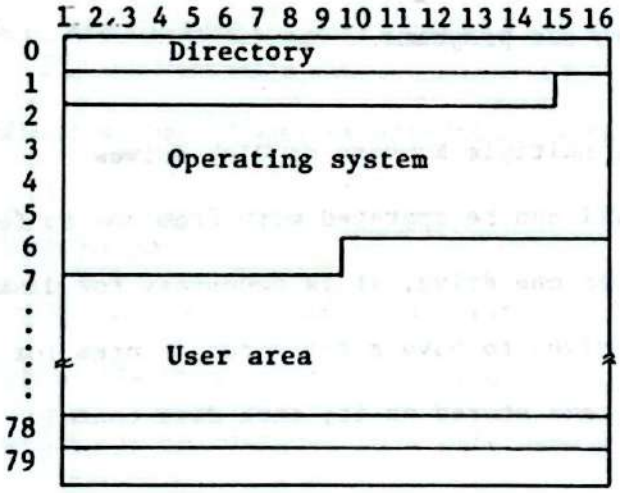
Data files and program files are stored in combinations on floppy disks. Memory areas of floppy disks are divided into four sub-areas as shown below.

1) Floppy disk


Resident area
File directory area
for the essential or resident part of OS
Area used for other parts of OS
for registration of names of files and programs
User area
for the user's data file and program files

2) Layout of M20/M23 floppy disk contents

Track/sector



Each sector consists of 256 bytes and is also called a block. Each track consists of 16 sectors and, therefore, each track consists of 4,096 bytes (4K bytes). A single floppy disk consists of 80 tracks, totalling 327,680 bytes (approximately 330K bytes).

1.7.3 Naming Floppy Disk Data Files

When a data file is read from or written onto a disk, the operating system handles the file by its name only and disregards its location.

File names

The rules for naming files are:

a) All 36 alphanumeric characters A through Z and 0 through 9 can be used for the file name.

b) Each file name can consists of up to 6 alphanumeric characters.

For example, names ABC1, FILE1 and 1F2B are all acceptable, but FILE1234 and @TANK are illegal names, since the former exceeds 6 characters and the latter contains the symbol @ which is not an alphanumeric character.

File names SOUTA, SOUTB and POUT are reserved names and should not be used for data files and programs.

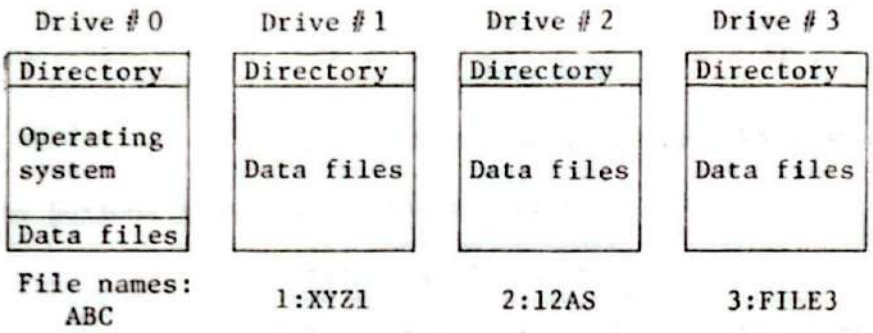
1.7.4 File Names for Multiple Numbers of Disk Drives

The M20/M23 Mark III can be operated with from one to four drives. When using more than one drive, it is necessary for disks in the second, third and fourth drives to have a registration area for the names of the files and programs stored on it; each disk contains its own file directory.

Note that if each disk has the names of files it stores, the Operating System manages all the files in all disks.

Format for file names including drive numbers

It is also necessary for any file on a disk in drive #1, #2 or #3 to contain the number of its drive at the beginning of its own name. This enable



If there are four different files with same names, say ABC, on four different disks in four drives, they can be distinguished by their drive numbers. They would properly be named ABC, 1:ABC, 2:ABC, and 3:ABC.

Format

```
[(drive number):] file name

Note:  Drive number 1, 2, or 3 should be written, but 0 can be
       omitted for drive #0.

Examples:  ABC, 1:XYZ1
```

[ ] indicates that it may be entered as needed.

1.7.5 File Name Extension

Each file name can be followed by a period (.) and from one to three alphanumeric characters. This is called a file name extension. This extension can be assigned to a file name either by the user or, if unspecified, by the operating system. The extension generally indicates the file's format. Extensions assigned and recognized by the operating system help the user and the system to identify file formats. Two extensions the user should know right away are .SAV and .APU.

Extension

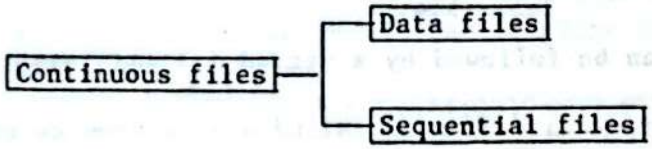
Format	File name.XXX
file name .SAV	Indicates a save file. This is run under the direct control of the operating system.
file name .APU	Indicates that the file is written in intermediate language and is run in BASIC mode.

Remember that a file name extension in an integral part of the official file name. Thus, the name used to reference file ABC .APU is not ABC but ABC .APU.

1.7.6 Organization of Disk Files

The operating system allocates data files and programs in two ways customarily called file organization methods. Files stored on disks in these ways are divided into continuous files and sequential files.

File classification according to organization:



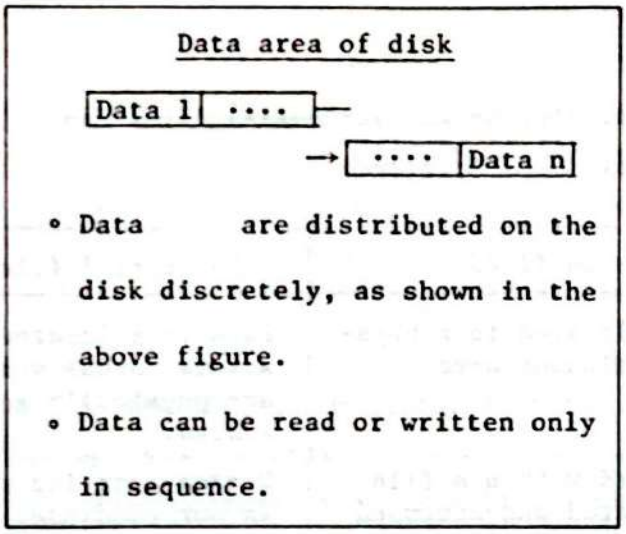
1) Continuous files

Data area of disk

Data 1 ..... Data n

- Data is allocated a continuous area on the disk.
- It's possible to exchange and correct data within the file.

2) Sequential files


Data area of disk
Data 1
····
····
Data n
• Data are distributed on the disk discretely, as shown in the above figure.
• Data can be read or written only in sequence.

3) The characteristics of continuous and sequential files may be summarized as follows:

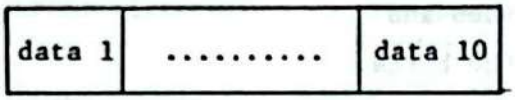
	Continuous files	Sequential files
Allocation of area	Data is allocated in a physically continuous area.	Data is allocated in several areas which are physically separated.
Correction of data	Every record within a file can be located and accessed directly for modification.	Random updating of data is not possible.
Speed of input and output	File input and output is very rapid since all data is in a continuous area.	File input and output speed depend way in which file area on the disk.
Efficiency in using disks	The user should reserve an area of pre-determined size for the file. This makes handling this type of file less troublesome.	A sequential file can make full use of disk space because the operating system searches for vacant areas on the disk and allocates them to the file.
Knowledge of the file size	The user should have a good idea of the size of the file beforehand.	It is not necessary for the user to estimate the file's size.
Applications	Suitable for files of known size and those which will be modified later.	Suitable for files used in scientific and engineering calculation

1.7.7 Disk File Management

The operating system manages files by their names.
This section describes how each file is allocated its area on the disk.

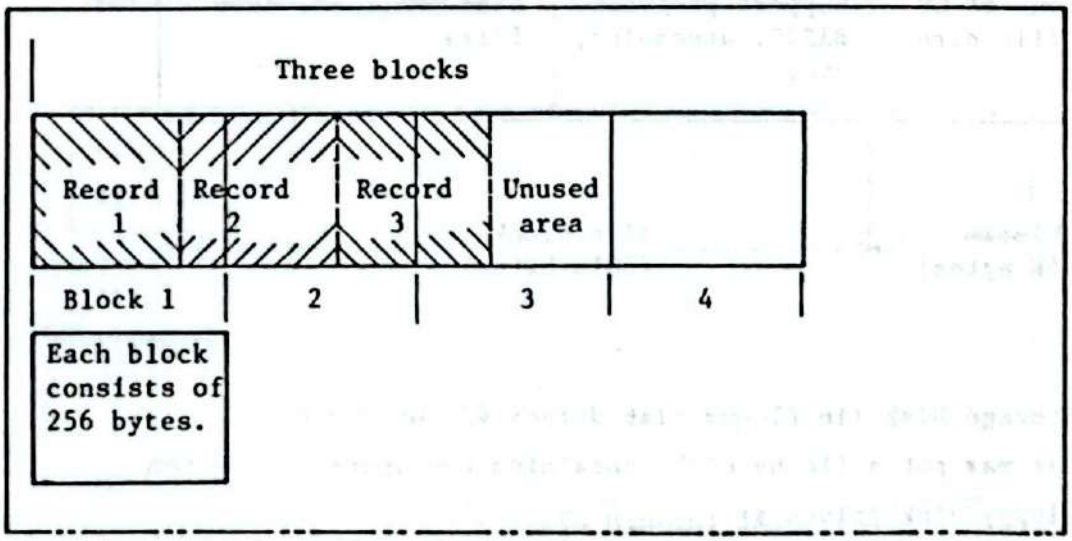
The minimum unit of space in which the operating system manages disk areas is called a block. Each block consists of 256 bytes.

As mentioned in the previous section, a continuous file is stored or retrieved in units of records.
These records are collections of data items, ten in the following examples:



..... a record of 200-byte length

One record may consist of only one data item.
Let us consider that a given file consists of three 200-byte lengths. Since the file needs 600 bytes, this file requires three blocks (=256 bytes x 3=768 bytes).

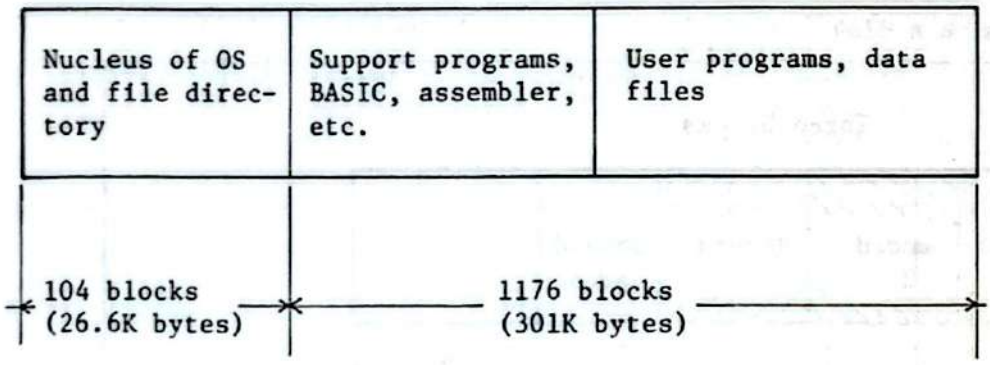


The record length of a data file should be given to the operating system before the file is written onto disk.
The record can be of any length and has nothing to do with the block size.
The operating system will assign an area to the file in the most economical way.

1.7.8 Floppy Disk for M20/M23

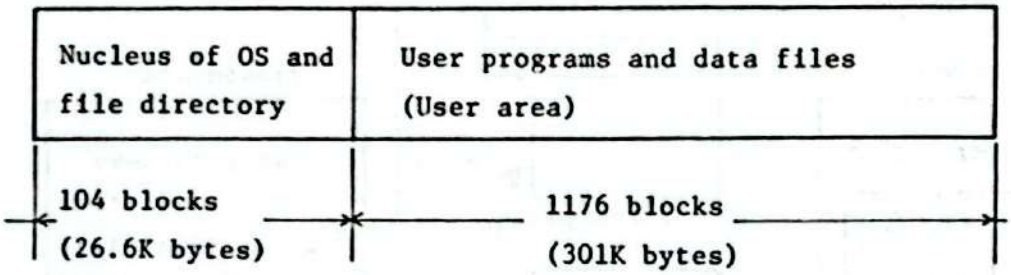
1) The capacity of a floppy disk (5 1/4 inch mini floppy disk) is referred to using the term single-sided double density.
   Each block consists of 256 bytes and
   1,280 blocks results in 327,680 bytes.

2) System disk (in floppy disk drive #0)
   The operating system, which consists of the nucleus and support programs, the file directory and other programs such as BASIC and FORTRAN IV, are stored on the disk in floppy disk drive #0)
   The remaining area is available for user programs and data files.

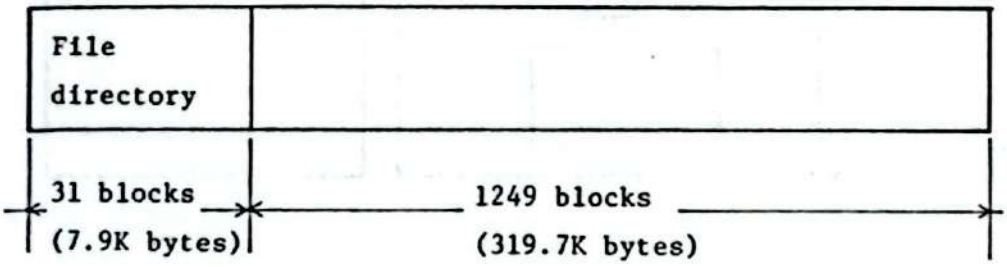

Nucleus of OS and file directory
Support programs, BASIC, assembler, etc.
User programs, data files
104 blocks (26.6K bytes)
1176 blocks (301K bytes)

3) Data Storage Disk (in floppy disk drives #1 through #3)
   The user may put a floppy disk containing the operating system into floppy disk drives #1 through #3.

The operating system transferred to the residential area of main memory from the disk in floppy disk drive #0 will be in control of the M20/M23 computer system.
In this case, the area occupied by the operating system on the disk in floppy disk drive #1 will be a dead area.
Thus, the following method of allocating area is recommended in order to make the best use of disk memory and accomodate user programs.



That is, on disks in floppy disk drives #1 through #3, 301K bytes of memory are available for user programs.
You may even drop the nucleus of the operating system from your disk in any drive. You can then use 1249 blocks of memory for your programs.

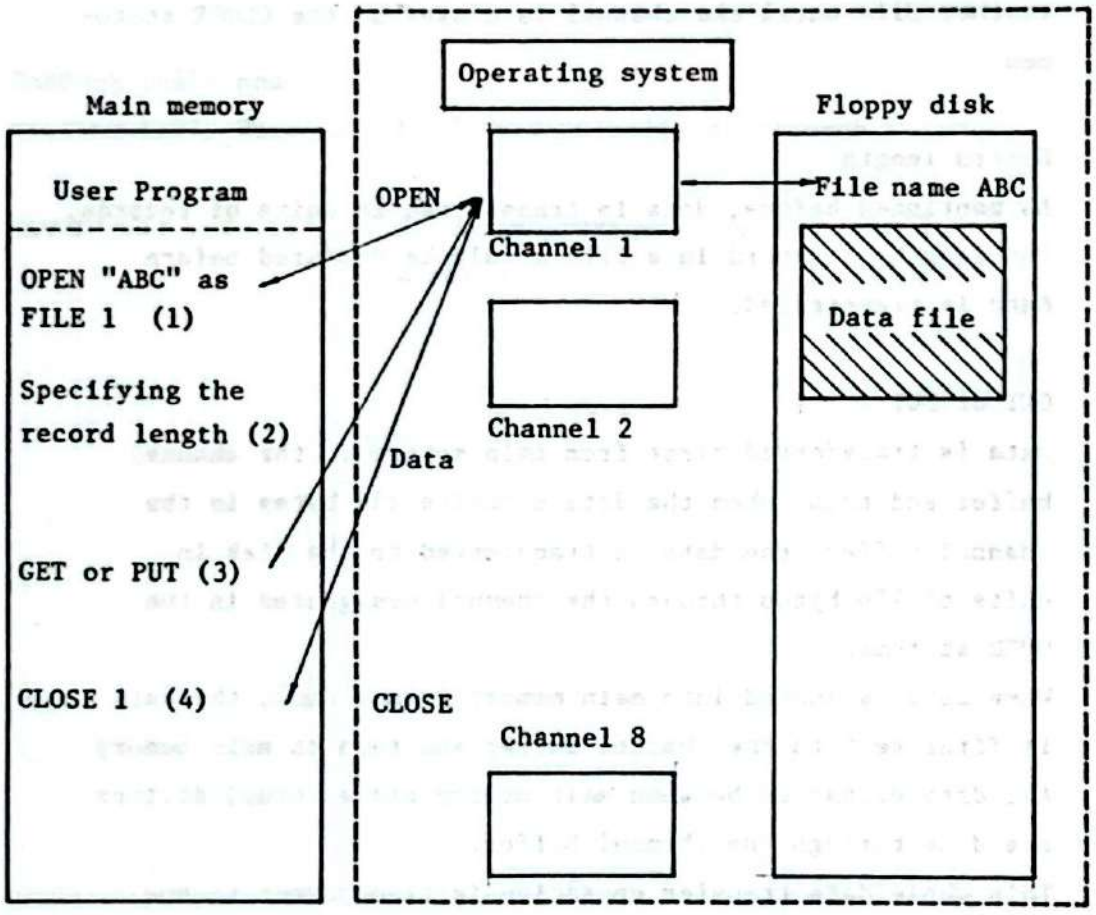


The last type of disk can be created with the INIT command.
It should be noted that a disk of this type should not be mounted onto floppy disk drive #0.

Above mentioned figures are available as of Rev.23A.

1.8 Data Transfer between Main Memory and External Devices via Channel

Data is transferred between main memory and external devices through channel buffers in main memory.
When writing a program, the user needs to specify the channel through which data will be transferred.



The diagram above is an example of a user program handling a data file on a floppy disk.

In this example:

1 OPEN and CLOSE

The OPEN statement opens channel 1 for access to a file named ABC. The user must specify, in the OPEN statement, the channel number with the file name under which the file is stored on the disk. This statement is used to give correspondence between channel 1 and the file name. Once this relation is established, this channel cannot be used to access another file until the channel is closed by the CLOSE statement.

2 Record length

As mentioned before, data is transferred in units of records. The length of record in a file should be declared before data is transferred.

3 GET or PUT

Data is transferred first from main memory to the channel buffer and then, when the data occupies 256 bytes in the channel buffer, the data is transferred to the disk in units of 256 bytes through the channel designated in the OPEN statement.

When data is loaded into main memory from a disk, the data is first sent to the channel buffer and then to main memory. All data exchanges between main memory and external devices are done through the channel buffer.

This whole data transfer operation is transparent to the user. This process is repeated until all data on a given file is transferred to its destination.

4 CLOSE

By executing a CLOSE statement, the channel specified in it will be closed to or released from the specified file.

Up to 256 bytes of data stored temporarily in the channel are closed.

OPEN	Specifying a file name and channel number
Specifying record length	Each channel consists of eight sub-channels numbered 1 to 8. They can be used concurrently by one program.
Reading (GET) and writing (PUT) data	Performed in units of records
CLOSE	Releases linkage between a file and a channel.

1.9 Sector Sequence

Data is stored on floppy disks in units of sectors (each sector or block consists of 256 bytes) and sectors are not physically linked with each other.
Because floppy disks are used with single read and write heads, sectors are arranged in the best sequence available for high-speed processing.

Sector sequence on a floppy disk

A / B	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
2	1	3	5	7	9	11	13	15	2	4	6	8	10	12	14	16
3	1	4	7	10	13	16	2	5	8	11	14	3	6	9	12	15
4	1	5	9	13	2	6	10	14	3	7	11	15	4	8	12	16
5	1	6	11	16	2	7	12	3	8	13	4	9	14	5	10	15
6	1	7	13	2	8	14	3	9	15	4	10	16	5	11	6	12
7	1	8	15	2	9	16	3	10	4	11	5	12	6	13	7	14
8	1	9	2	10	3	11	4	12	5	13	6	14	7	15	8	16

A ... Physical sector B ... Sequence

M20/M23 Mark III uses sector sequence 8 (As of September, 1981).

2. OPERATING SYSTEM

The functions of the M20/M23 Mark III operating system were outlined in Part 1.
This section details these functions, which are indispensable to the operation of M20/M23, and explains their applications.

These functions of the operating system are grouped together as the Command

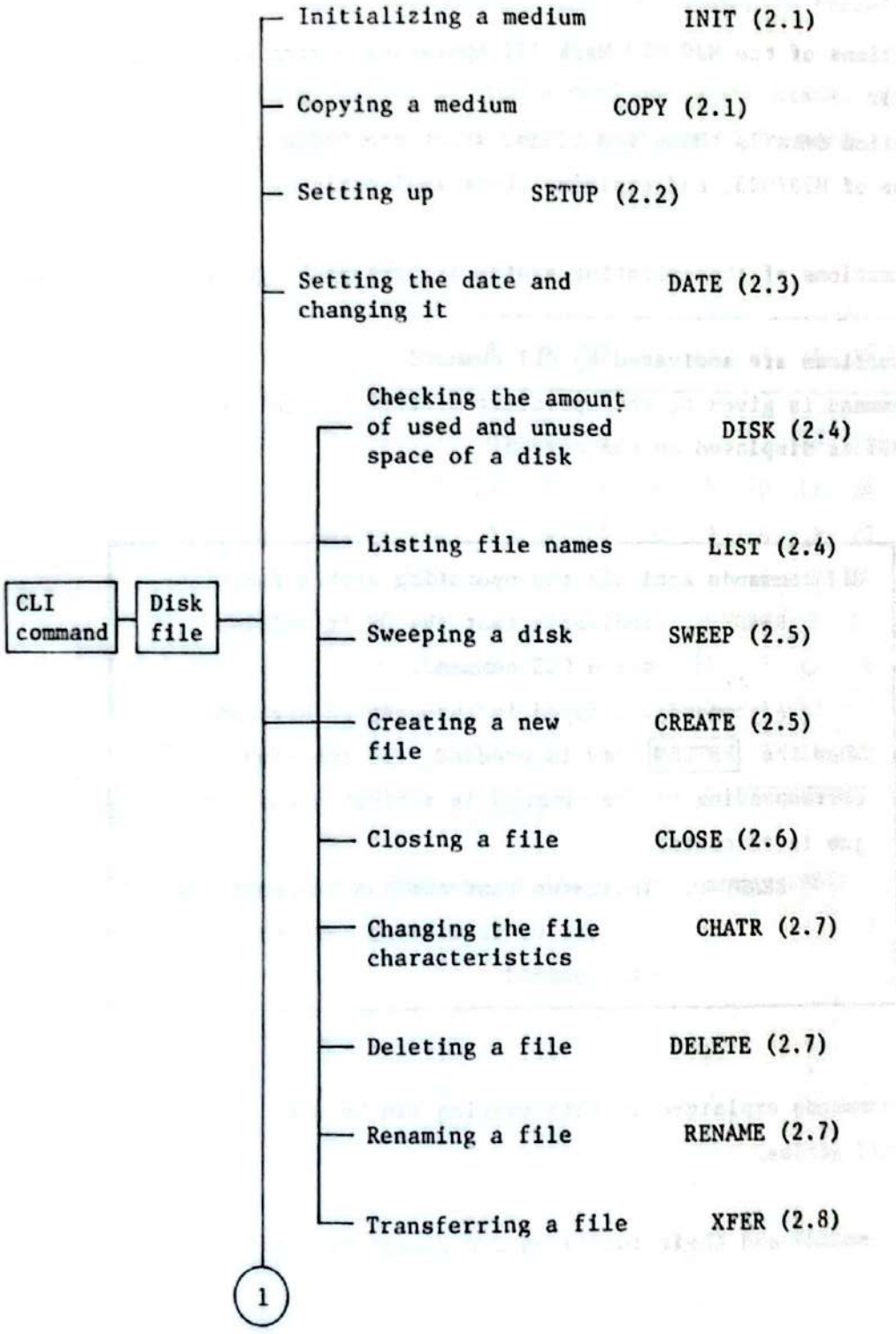
These functions are activated by CLI commands.
A CLI command is given to the operating system through the keyboard when READY is displayed on the screen.

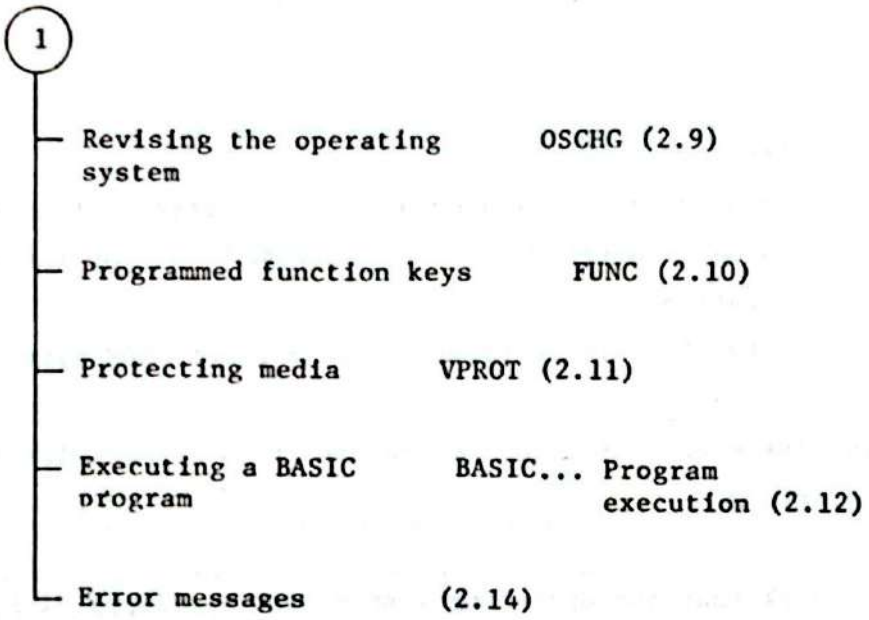
> CLI commands activate the operating system functions.
>
> READY ... indicates that the OS is waiting for a CLI command.
>
> A command ... typed in through the keyboard.
>
> When the [RETURN] key is pressed, the function corresponding to the command is activated and the job is executed.
>
> READY ... indicates that the job is completed and the OS is waiting for the next CLI command.

The CLI commands explained in this section can be used in both the M20 and M23 series.

The CLI commands and their functions are summarized in the following table.

The number following each command indicates the section in this manual in which the command is explained in detail.


CLI command
Disk file
Initializing a medium INIT (2.1)
Copying a medium COPY (2.1)
Setting up SETUP (2.2)
Setting the date and changing it DATE (2.3)
Checking the amount of used and unused space of a disk DISK (2.4)
Listing file names LIST (2.4)
Sweeping a disk SWEEP (2.5)
Creating a new file CREATE (2.5)
Closing a file CLOSE (2.6)
Changing the file characteristics CHATR (2.7)
Deleting a file DELETE (2.7)
Renaming a file RENAME (2.7)
Transferring a file XFER (2.8)
1


1
Revising the operating system
OSCHG (2.9)
Programmed function keys
FUNC (2.10)
Protecting media
VPROT (2.11)
Executing a BASIC program
BASIC... Program execution (2.12)
Error messages
(2.14)

2.1 Initializing Disks and Copying Disks

2.1.1 Disk Initialization

External memory is supplied by mini-floppy disks 5 1/4 inches diameter from Soft Sector Media "Verbatim MD550-01" of Information Terminals Corporation.
The capacity of a single floppy disk is about 350K bytes.

A new disk must be initialized before it is loaded with new data file and programs.

1 Check that the operating system disk in floppy disk drive 0 contains the INIT program.
In this section, as in other sections, it is assumed that the resident part of the operating system has already been loaded into main memory.

2 Mount a new floppy disk onto one of floppy disk drive 1 through floppy disk drive 3 and check that it is correctly in place.

3 After READY appears on the display screen, type in the CLI command word INIT, the slash mark (/) and the drive number in that order.

Upon pressing the [RETURN] key, the disk will be initialized and READY will reappear on the display screen.

```
CLI command: INIT
    function: Disk initialization
INT/n [RETURN]
            n  : One of floppy disk drive 1 through
                 floppy disk drive 3
```

B : Initialization of a disk in floppy disk drive #0

Now you have a new disk which is ready to be loaded with new data files or programs.
This whole process is called disk initialization.

Note 1:
If your system has only one drive, we recommend that you obtain new disks containing the operating system from a SORD agent.

Note 2:
You may put an old floppy disk onto one of floppy disk drive #1 through floppy disk drive #3 and initialize it.
Remember that initialization will destroy all data on the disk.

2.1.2 Copying Disks

A disk for the operating system can be prepared by copying the contents of the disk in floppy disk drive#0.
Remember that the disk with the operating system has a user area of 1176 blocks or 301K bytes in addition to the OS area and the file directory area.

The procedures for copying area are as follows:

1 Check to confirm that the operating system on the disk in floppy disk drive #0 has the COPY program.

2 Mount a new, initialized floppy disk onto one of floppy disk drive #1 through disk drive #3 and check that it is properly set.

3 After READY appears on the screen, type in the CLI command word COPY, the slash mark (/), and the drive number in that order.

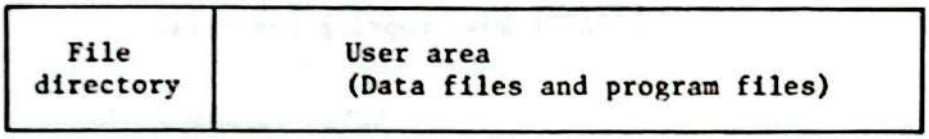
4 The copying function of the operating system will read out the contents of the disk in floppy disk drive #0, write it on one of the other disks, and compare the copy with the original.
If an error occurs, an error message will be displayed on the screen.
The user should then take appropriate action as listed in the section on error messages.

```
CLI command: COPY
    function: Copies the contents of a disk in
              floppy disk drive #0 to a disk in
              one of floppy disk drive #1 through
              floppy disk drive #3.
    A       : Indicates that the entire contents
              of a disk have been unconditionally
              copied.
              When /A is not specified, only
              used tracks are copied.
    B       : Used in the same manner as when
              entering COPY 0 - > 0 and press-
              ing the [RETURN] key.
```

2.1.3 Data Files

The user can delete some components of the operating system from a system disk to reserve a larger user area on the disk.

The user can also initialize a system disk (by means of the INIT command) to use the entire disk except for its file directory as a data storage disk.
The format of an initialized disk is illustrated below:



2.1.4 Extended COPY Functions

If an error occurs, when the COPY command is used, processing terminates.
The extended COPY command continues processing.
It handles errors by sectors and the contents of sectors remain as they were before copying.

The COPY command must be entered as follows:

COPY n_1 → n_2 [RETURN]

The contents of the disk in floppy disk drive #n_1 are copied to floppy disk drive #n_2 when the COPY command is entered and the [RETURN] key is pressed.
n_1 and n_2 may be from 0 to 3 and n_1 may be equal to n_2.

When n_1 or n_2 is specified by 0, the following message appears on the screen.
The user must press the [SPACE] key after changing the disk.

(Example 1)

```
COPYΔ0->1 [RETURN]
     SET COPY DISKETTEΔ
```

Mount the disk to be copied and press the [SPACE] key.

SET OS DISKETTEΔ (after copying)

Mount the disk containing OS onto drive #0 and press the [SPACE] key.

READY

When the [LF] key is pressed instead of [SPACE] key, copying restarts.

(Example 2)

COPYΔ1->2 [RETURN]

Copying starts

READY

When n_1 is equal to n_2, the following message appears on the screen. The user can copy the contents of a disk only after changing the disk.

Ten tracks are copied each time a disk is exchanged.

(Example 3)

COPYΔ0-1>0 [RETURN]

SET SOURCE DISKETTEΔ

Mount a disk, the contents of which are copied onto the drive#0 disk and press the [SPACE] key.

SET DESTI DISKETTEΔ

Mount another disk, the contents of which are also copied onto a drive #0 disk and press the [SPACE] key.

2.2 Setup

The M20/M23 Series provides a set-up function to determine the state of bootstrapping.
Conditions defined during set-up can be stored.

CLI command: SETUP

function: Defines when the system starts and the state of SIO

SETUP[/D] [RETURN]

/D ... indicates that set-up starts with the standard value defined at loading time.

Keys used for set-up and their functions

[SPACE] : Sets the next value in place of a current set-up item. The current contents of a set-up item are displayed in reverse mode.

[RETURN] : Terminates the current set-up item and moves to the next item.

[ESC] : Does not change the contents of the current set-up item and returns to OS.

[↑] [↓] : Controls the vertical synchronous position.

[←] [→] : Controls the horizontal synchronous position.

[W] : This key is used to store the contents of definitions used for the next system generation on a disk.

[E] : Completes the set-up operation and returns to OS.

Set-up items are classified into two categories:

- Items used for determining environments of a display or keyboard
- Items used for determining characteristics of SIO

Items for determining M20/M23 operating environments are:

- LINE/SCREEN
- COLUMN/LINE
- HORIZONTAL SYNC. POSITION
- VERTICAL SYNC. POSITION
- CHARACTER COLOR
- SCREEN COLOR
- ERROR MESSAGE LETTER
- TYPE HEAD
- AUTO REPEAT START TIME
- AUTO REPEAT INTERVAL TIME
- BUZZER-1
- BUZZER-2

Items for determining SIO characteristics are:

- BAUD RATE
- STOP BIT
- BIT/CHARACTER
- PARITY
- IGNORE

Any value changed during set-up cannot be used during the set-up. Neither can it be used when the set-up is terminated by the E key.

SET-UP

ENVIRONMENT

LINE/SCREEN		25	SIO A CHANNEL	TABLE
COLUMN/LINE		80		
HORIZONTAL		92 (↔)	BAUD RATE	9600
VERTICAL		28 (↑↓)	STOP BIT	2
CHARACTER COLOR		GREEN	BIT/CHARACTER	8
SCREEN COLOR		BLACK	PARITY	OFF
ERROR MESSAGE LETTER		ALPHABET	IGNORE	OFF
TYPE AHEAD		ON		
AUTO REPEAT START TIME		400ms	SIO B CHANNEL	TABLE
AUTO REPEAT INTERVAL TIME		60 ms		
			BAUD RATE	9600
	LENGTH	TONE	STOP BIT	2
BUZZER-1 ON	48	176	BIT/CHARACTER	8
BUZZER-2	12	144	PARITY	OFF
			IGNORE	OFF

2.3 Setting the Date

The operating system has a dating function.
If the user sets the date when the operating system is loaded into main memory, then all user programs will be tagged with that date automatically. Therefore, the user need not issue the DATE command to the operating system.
If the user did not set the date initially, or if the wrong date has been set, the correct date can be entered by using the DATE command.

> CLI command: sets the date or changes it
>
> DATE DD-MM-YY
>
> The day, month and year must be typed in the above order, every two digits separated by hyphens.

(Example)

This example sets the date of September 15, 1981.

```
READY
DATE   15-9-81
READY
```

This date entry can be made as illustrated below;

1	5	-	9	-	8	1	RETURN

or

1	5	-	S	E	P	-	8	1	RETURN

2.4 Checking Unused Areas and Contents of Disks

2.4.1 Checking the used and unused areas

Before storing a data file or program on a floppy disk, the user needs to know the size of the unused area on the disk.
The DISK command does this job.

```
CLI command:  DISK
    Function:  Checks the amount of used and unused
               space on the disk.
    DISK[/n] [RETURN]
    LEFT:xxx  USED :xxxx
           n ... specifies one of floppy disk drives
                 #0 through #3.
                 n can be omitted only when floppy
                 disk drive #0 is specified.
        LEFT ... Number of unused blocks
        USED ... Number of used blocks
                 Each block is 256 bytes long.
```

```
READY
DISK/1  ..............  Checking the used and unused
           .                area
LEFT:  982  USED:  298

READY
```

2.4.2 Listing File Names

When the user wants to see what is stored on a disk in any drive, the list of names of all files stored on the disk can be called out with

the LIST command.

CLI command: LIST

Function: Lists the names of files stored on the disk.

LIST[/n][/A][ΔSOUTA[/m] RETURN

n ... specifies one of floppy disk drives #0 through #3.
n can be omitted only when floppy disk drive #0 is specified.

/A ... specifies the entire contents of the disk, including the operating system. If you want only a list of the user programs, this option can be omitted from the LIST command.

ΔSOUTA ... prints the list on the printer. Δ represents one space. If you want to see the list on the display screen, SOUTA can be omitted from the LIST command.

/m ... specifies the type of the printer to be used. m is called the printer mode, details of which are in the M20/M23 Business Handy Computer hardware manual.

(NOTE 1)

Brackets represent an option which may be omitted, depending on the situation.

(NOTE 2)

[ΔSOUTA[/m]] ... Output to the printer.

ΔSOUTA .... Output to SIO channel A

ΔSOUTB .... Output to SIO channel B

ΔPOUT .... Output to POUT

m stands for the printer mode.

(NOTE 3)

Δ represents a space key entry.

Some CLI commands may not be processed if this key is not used correctly.

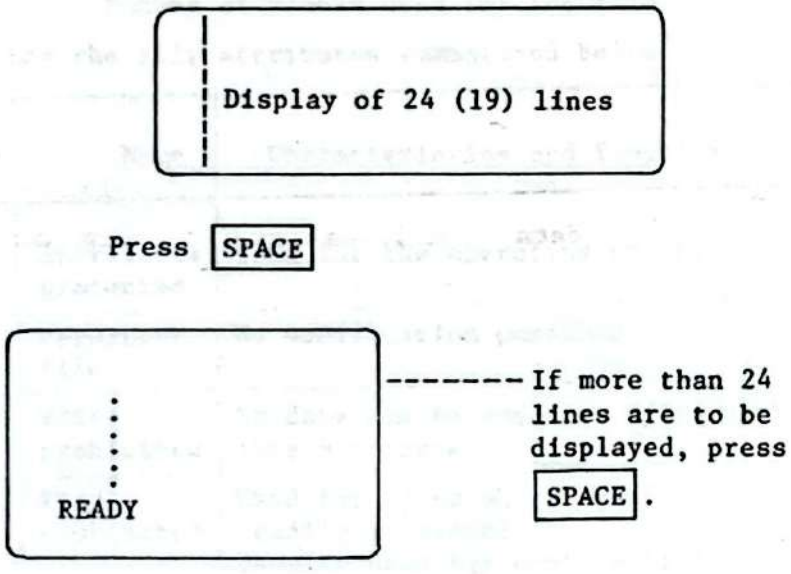
Mount the disk containing the operating system onto floppy disk drive #0 and enter the LIST command as illustrated below:

LIST/A [RETURN]

The list on the screen has only 24 (19) lines.

If you want to see the rest of the list, press the [SPACE] key.

The value 24 (19) indicates that each display screen contains 80 columns and 20 lines.

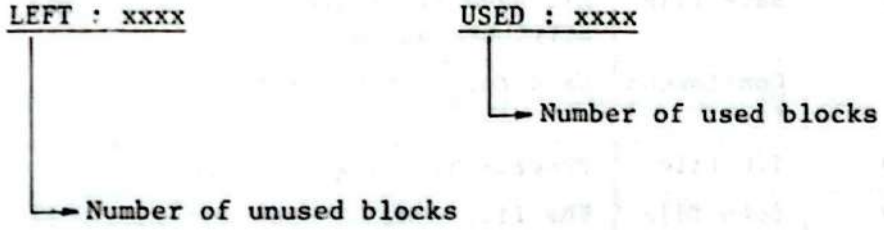


The list is printed by the following command:

```
READY
LIST/A     SOUTA/3

READY
```

(1) The top of the list will be the same line as is displayed by the DISK command.



(2) The format of the file names listed in this example is as follows:

```
xxxxxx. YYY nnn xxxxx xx-xxx-xx xxx
  a          b   c       d      e
```

a. xxxxx ..... File name
   YYY ..... File extension
   .SAV .... indicates an executable program
   .APU .... indicates a BASIC program.

b. nnn ..... Number of blocks used for the file

c. Indicates the file attributes summarized below.

Attribute	Name	Characteristics and function
A	Attributes protected	Used for the operating system
P	Permanent file	No modification possible
W	Write prohibited	No date can be added to files with this attribute
R	Read prohibited	Used for files which should not be readily accessible. Usually used for confidential data or programs.
X	Transfer prohibited	Used for files which should not be transferred.

Attribute	Name	Characteristics and function
S	Save file	All save files are tagged with this attribute automatically.
C	Continuous file	Used for continuous files
B	I/O file	Program handling I/O files
O	Open file	The file remains open and should be closed with the CLOSE command before further handling.

d. xx-xxx-xx ... Date the file was created
The date was typed in the form of DD-MM-YY, for example 15-09-81, but is displayed or expressed as 15-SEP-81.

e. xxx ..... Version number of the operating system, for example 23D

The operating system of M20/M23 can contain up to 168 file names on a single disk.

2.4.1 Extended Switches for the LIST Command

The LIST command contains extended switches S and W.
Both can be used with switch A.

(1) Switch S (Sorting)

When the LIST command includes switch S, the file names are sorted for listing.

(Example)

LIST/S RETURN

LIST/A/S RETURN

(2) Switch W (Wild card function)

When the LIST command includes switch W, a question mark (?) and asterisk (*) can be used for file name specification.

? ... Indicates the existence of a single character.

* ... Indicates the existence of a number, including zero.

(Example)

When you need to list all the save files, you can enter the LIST command as illustrated below:

LIST/A/WΔ*. SAV RETURN

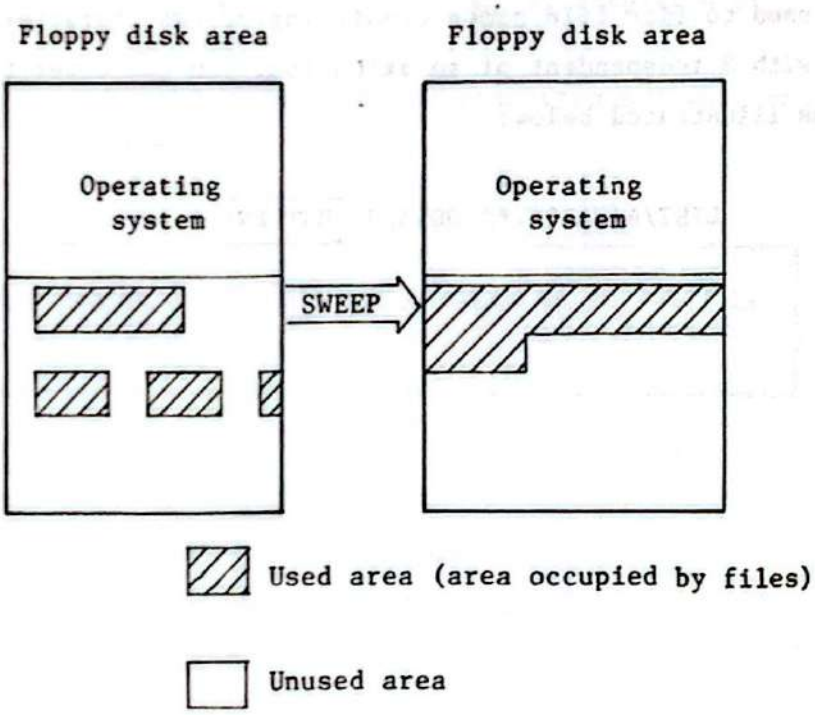
When you need to list file names consisting of two characters and starting with S independent of an extension, you can enter the LIST command as illustrated below:

LIST/A/WΔS?*.*ΔSOUTA/3 RETURN

2.5 Creating Disk Files

2.5.1 Sweeping Disk Files

When a disk has been used for a long time, it may happen that used and unused blocks are scattered randomly throughout it.
This means that a new continuous file may not be able to be created even if there are actually enough unused blocks as displayed by the DISK command, to accomodate the new file.
The operating system offers the SWEEP command to sweep the disk so that old user files are aligned continuously in the upper-most part of the user area on the disk.



CLI command: SWEEP

Function: Sweeps the used blocks.
All used blocks will be aligned continuously from the top of the user area.

SWEEP[/n] RETURN

n ... specifies one of floppy disk drives #0 through #3.
n can be omitted only when floppy disk drive #0 is specified.

(Example)

```
READY
SWEEP/1 ..... Sweeps all used blocks for the file.
SWEEP END

READY
```

2.5.2 File Creation

A new file can be created on the disk by a BASIC program as well as by the CREATE command.
Reservation of an area for a new continuous file, however, is possible only with the CREATE command.

1) Creating a continuous file

A continuous file can be created only on continuous blocks on the disk. Since the record file can be of any length, arranging it is relatively easy.

The user should check the following before starting:

(1) Are there enough blocks available for the file?
This can be checked with the DISK command.

(2) Are there enough continuous blocks for the continuous file?
Do you need to check with the SWEEP command?

(3) Is there any file on the disk which is needed for the new file?

CLI command: CREATE (a continuous file)

CREATEΔ[n:](file-name)/CΔn_1*m RETURN

n ... specifies one of floppy disk drives #0 through #3.
n can be omitted only when floppy disk drive #0 is omitted.

file-name ...
Name of the file to be created

/C .. specifies creation of a continuous file

n_1 .. Number of records

m ... Number of bytes in a record

(Example) Creating a continuous file named SALES
The file is supposed to have 50 records, each of which consists of 100 bytes.
The file is to be created with the name SALES on the disk in floppy disk drive #1.
Remember that the file needs 50 records of 100 bytes each or 5K bytes, and it should be verified that there is enough area for this file on the disk.

```
READY

LIST/1
LEFT : 876  USED : 484 .................  (1)
OTHELO.APU  57              1-SEP-81

READY
CREATE  1 : SALES/C 50*100 .............  (2)

READY
LEFT : 856 USED: 424 ...................  (3)
OTHELO.APU 57               1-SEP-81  00A
SALES      20C             15-SEP-81 .....  (4)

READY
```

(1) There are 876 unused blocks and this file needs 50 blocks. Thus there are clearly enough blocks available to create a new file.

(2) The CREATE command creates a file named SALES.

(3) This step checks that the SALES file has been created on the disk. Notice that there are 876 unused or remaining blocks, now 20 fewer than before.

(4) There is a new file named SALES occupying 20 blocks as a continuous file on the disk in floppy disk drive #1.

2) Creating a sequential file

A sequential file is created on physically separated areas on the disk. The user should check the following before starting:

(1) Are there enough blocks available for the file?

(2) Is there any file on the disk which is needed for

the new file?

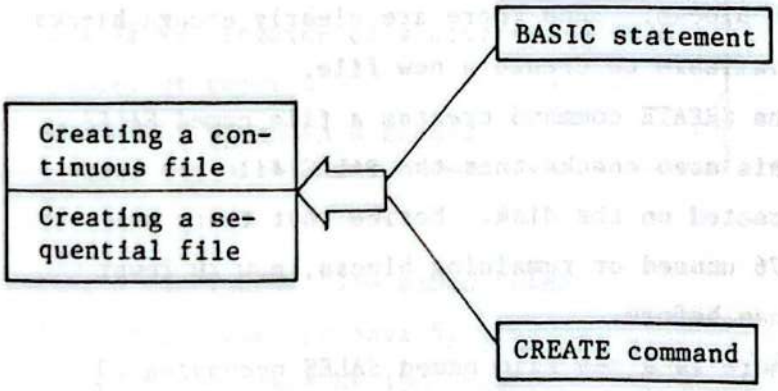
(3) Do you have a clear idea of the file's size?
(It is not necessary to know the exact number of blocks.)

CLI command: CREATE (a sequential file)

CREATE [n:](file-name) RETURN

n ... specifies one of floppy disk drives #0 through #3.
n can be omitted only when floppy disk drive #0 is specified.

file-name ...
Name of the file to be created



(Example) Creating a sequential file named SALES

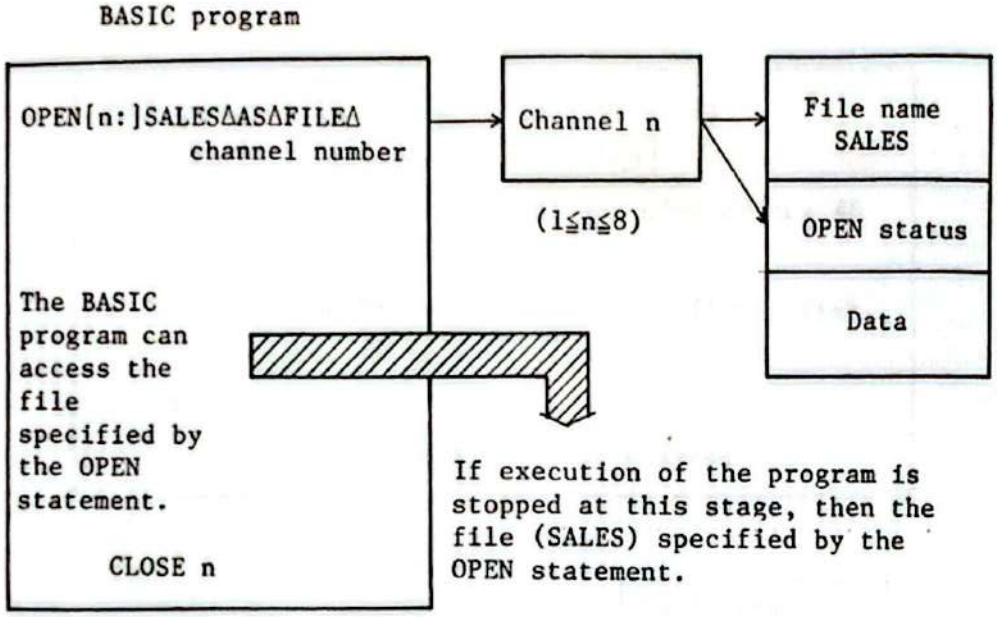
```
READY
CREATE     1: SALES ........................... (1)
READY
LIST/1
LEFT : 876  USED : 404
OTHELO.APU  57          1-SEP-81  00A
SALES                  15-SEP-81 .............(2)
READY
```

(1) The CREATE command creates a sequential file named SALES.

(2) This step checks that the SALES file has been created on the disk in floppy disk drive #1.

2.6 Closing Data Files

A data file must be opened by the OPEN statement before it is processed, or closed by the CLOSE statement after it has been processed.
The mechanism of data file handling is illustrated below:



A file is made accessible after the OPEN statement has been executed and made inaccessible after the CLOSE statement has been executed.
The file named SALES cannot be processed through channel n by the CLOSE statement.
The CLOSE statement should be executed before the program terminates.

If execution of a program is stopped after the OPEN statement has been executed and before the CLOSE statement is executed, then the file will remain in open status.
Such a file is tagged with file characteristics.
A file in open status cannot be accessed by the OPEN statement executed as part of another BASIC program.

If it is accessed by the OPEN statement, an error message will be displayed on the screen and the file may be damaged.

A file in open status can be closed by the CLOSE command.

CLI command: CLOSE

CLOSE [/n] RETURN

n ... specifies one of floppy disk drives #0 through #3.
n can be omitted only when floppy disk drive #0 is specified.
Any file on the disk in the drive will be closed by this command.

(Example)
A sequential file named SALES is found to be open when the file directory of the disk associated with floppy disk drive #1 is listed by the LIST command.
The file should be closed for reusing as illustrated below:

```
READY
LIST/1
LEFT:876 USED:484
OTHELO. APU  57              1-SEP-81
SALES         8 0           15-SEP-81  ..... (1)
READY
CLOSE/1  ..................................... (2)
READY
LIST/1
LEFT:876 USED:484
OTHELO. APU  57              1-SEP-81
```

```
SALES          8          15-SEP-81  .....(3)
READY
```

(1) The file characteristic 0 stands for open status of the file.

(2) All files which remain open will be closed by the CLOSE command.

(3) Note that file characteristic 0 has been dropped -- the file named SALES has been closed.

2.7 Renaming Files and Changing File Characteristics

2.7.1 Renaming Files

It may happen that the user wants to change the names of data files or programs which have been registered in the file directory.
The M20/M23 operating system provides a CLI command for just such jobs. Using the RENAME command, only the name registered in the file directory will be changed; the file itself will be left unchanged.

CLI command; RENAME

RENAMEΔ[n:](file-name)Δ

Renaming → [n:](file-name) RETURN

n ... specifies one of floppy disk drives #0 through #3.
n can be omitted only when floppy disk drive #0 is specified.
(The same n should be repeated.)

The old name will be replaced with the new name in the file directory of the disk.
Remember that the name extension is an indispensable part of the file name.

(Example)

This example renames the file named SALES on the disk associated with floppy disk drive 1 as SALESA.
If the file has an extension, it must be included in the file name.

```
READY
LIST/1
```

```
LEFT:876  USED:484
OTHELO. APU  57            1-SEP-81
SALES         8           15-SEP-81
READY
RENAME  1:SALES  1:SALESA
READY
LIST/1
LEFT:876  USED:484
OTHELO. APU   57           1-SEP-81
SALESA         8           15-SEP-81
```

2.7.2 Changing File Characteristics

New characteristics (or attributes) may be added to and old attributes may be dropped from a file to protect the file.
The following attributes can be given to a file by the user.

A	Attributes protected
P	Permanent file
R	Read prohibited
W	Write prohibited
X	Transfer prohibited

CLI command: CHTR

CHATRΔ[n:](file-name)[/A][/P][/W][/X] RETURN

n ... specifies one of floppy disk drives #0 through #3.
n can be omitted only when floppy disk drive #0 is specified.

file-name ...
specifies the name of a program or data file whose characteristics are to be changed.

[/A] ... Attributes to be added

(NOTE 1) The CHATR command is invalid for files with the A attribute, and all attributes of the file are protected.

(NOTE 2) No data file or program with both P and A attributes can be deleted from the disk.

(NOTE 3) Once attribute X is assigned to a file, it cannot be deleted by the CHATR command.

(Example)

This example changes the characteristics of a file named SALES.

```
READY
LIST/1
LEFT:876  USED:404
OTHELO. APU   57          1-SEP-81
SALESA         0         15-SEP-81
READY
1:SALESA/W
READY
LIST/1
LEFT: 876  USED:404
OTHELO. APU   57          1-SEP-81
SALESA         0  W       15-SEP-81
READY
```

2.8 Deleting and Transferring Files

2.8.1 Deleting Files

An old program or data file which is not needed anymore can be deleted from a disk by the DELETE command.

CLI command: DELETE

DELETEΔ[n:](file-name) RETURN

DELETEΔ[n:](file name)Δ[n:](file-name) ... RETURN

n ... specifies one of floppy disk drives #0 through #3.
n can be omitted only when floppy disk drive #0 is specified.

file-name ...
specifies a program name and data file name, including any extension.

(NOTE 1) More than one file can be deleted with a single DELETE statement.

(NOTE 2) A file with the P attribute cannot be deleted unless the attribute is dropped.

(NOTE 3) If a file remains in open state, it cannot be deleted by the DELETE command. This kind of file should be closed by the CLOSE command before the DELETE statement is used.

(Example)

This example deletes a file named SALESA stored in the disk associated with floppy disk drive

```
READY
LIST/1
```

```
LEFT: 876    USED: 404
OTHELO.APU          57          1-SEP-81
SALESA               0  W       15-SEP-81
READY
DELETE L:SALESA
READY
LIST/1
LEFT: 876    USED:404
OTHELO.APU          57          1-SEP-81
READY
```

2.8.2 Transferring Files

It may happen that part of the file is damaged during processing. Therefore, it is recommended that a copy of the file be kept on another disk.

XFER is the CLI command used to copy files onto other disks. The copied file is customarily called a back-up file or simply a back-up.

No file with attribute X can be transferred by the XFER command. The two types of files, continuous and sequential, are each transferred slightly differently.

CLI command: transfers a file to be written as a continuous file

(1) Program transfer

XFERΔ[n_1:](program-name)Δ[n_2:](program-name) RETURN

n_1 ... The number of the drive holding the disk with a file to be transferred.

n_2 ... The number of the drive holding the

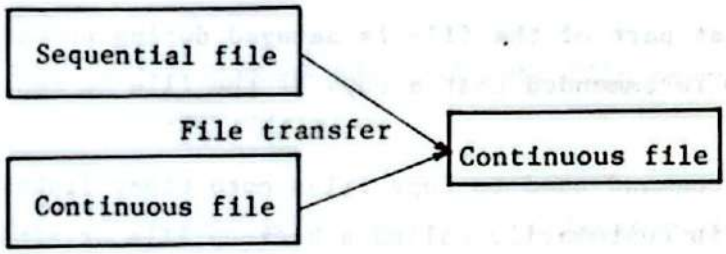
disk containing transferred file.

n .... specifies one of floppy disk drives #0 throught #3.
n can be omitted only when floppy disk drive #0 is specified.

program-name ...
A program name must include its extension.

The transferred file is stored in a continuous file.

(2) Transferring a file to be stored as a continuous file



Remember that an area of continuous blocks should be reserved for the file with the CREATE command.
It is recommended that program files be stored as continuous files.

XFERΔ[n_1:](file-name)Δ

Transfer [n_2:](file-name) RETURN

n_1 ... The number of the drive holding the disk with a file to be transferred.

n_2 ... The number of the drive holding the disk containing the transferred file.

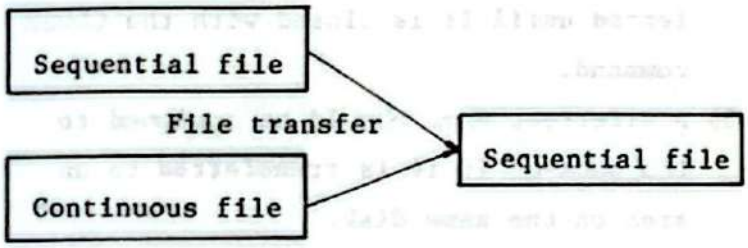
n .... specifies one of floppy disk drives #0 through 3.

n can be omitted only when floppy disk drive #0 is specified.

file-name ...

A continuous file must be reserved on the disk containing the transferred file.

(3) Transferring a file to be stored as a sequential file



When a file is transferred to a sequential file within the same disk, a new name should be assigned to the new file.
As in cases (1) and (2), the same name is transferred to a disk in a different drive.

XFERΔ[n_1:](file-name)Δ

Transfer [n_2:](file-name) RETURN

n_1 ... The number of the drive holding the disk with a file to be transferred.

n_2 ... The number of the drive holding the disk containing the transferred file.

n .... specifies one of floppy disk drives #0 through #3.
n can be omitted only when floppy disk drive #0 is specified.

file-name ...
Any extension used must be included in the file name.

(NOTE 1) A continuous area should be reserved for continuous files with the CREATE command.

(NOTE 2) A file in open status cannot be transferred until it is closed with the CLOSE command.

(NOTE 3) A diferrent name should be assigned to the back-up if it is transferred to an area on the same disk.
If it is transferred to another disk, the same name may be used.

(Example)

This example transfers the OTHELO.APU program on the disk in drive #1 to a different file area on the same disk by remaining it GAMES.APU.

```
READY
LIST/1
LEFT: 876   USED:404
OTHELO.APU  57                  1-SEP-81
READY
XFER 1:OTHELO.APU       1:GAMES.APU
READY
```

```
LIST/1
LEFT: 819  USED: 461
OTHELO. APU  57              1-SEP-81
GAMES. APU                   15-SEP-81
READY
```

2.9 Loading a New Revised Version of the Operating System

SORD always endevors to offer the best software and hardware to its customers.

For this reason, we have released various operating systems.

When SORD releases a new revised version of the operating system, SORD computer users may be asked to replace their old versions with a new one.

This replacement of the operating system is done in one of the following ways;

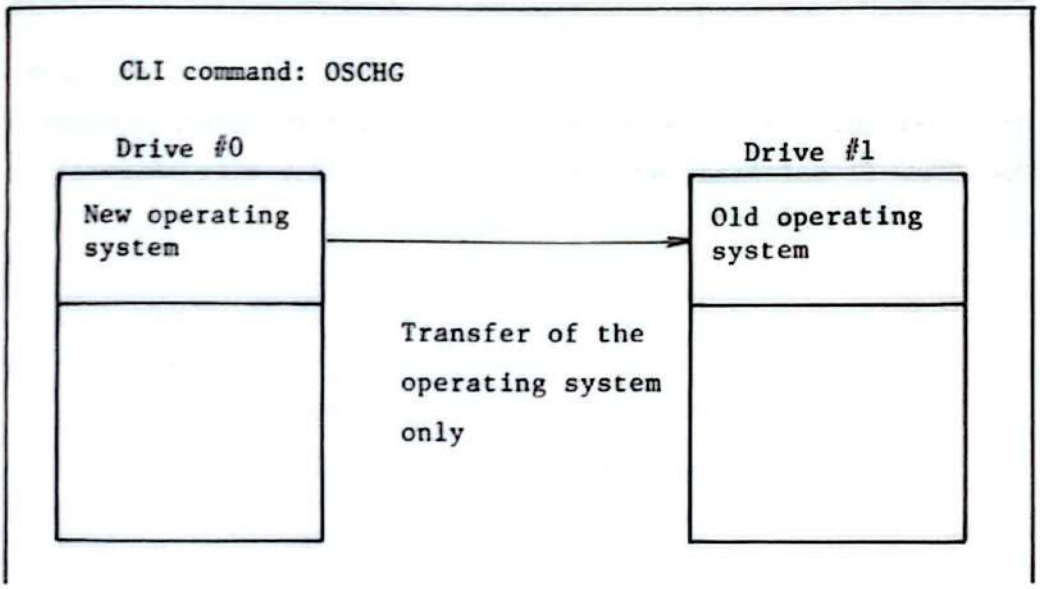
1. Copying

   This involves copying the new operating system onto the disk which holds the old version.

   Since all programs and files on the disk will be lost together with the old operating system, this method is recommended only when the old programs and files are not being used any more.

2. Replacing only the operating system

   If the user wants to keep the old programs and files, merely the old operating system can be replaced with the new version. This can be done with the OSCHG command.



```
READY
OSCHG

      OS COPY  0 - > 1 [RETURN] ..... A copy of the
                                      the new operat-
                                      ing system
                                      starts.
      OS COPY  0 - > 1 ..... One copy of the new
                                operating system is completed
                                and another copy of the
                                new operating system
                                starts.
      OS COPY  0 - > 1 ..... Press the [RESET] key
                                to terminate copying.
```

Procedure notes:

(1) To replace the old version with the new one, you must set the disk with the new version of the operating system on floppy disk drive #0 and a disk with the old version in floppy disk drive #1.

(2) If you want to make more than one copy of the new version, then instead of pressing the [RESET] key when so instructed by the message OS COPY 0 - > 1 displayed on the screen, you must replace the disk in floppy disk drive #1 with another disk containing the old version and press the [RETURN] key.
This procedure can be repeated as often as you like.

(3) When the OSCHG job has been completed by pressing the [RESET] key, the M20/M23 will become inactive and receive no further instructions.
To reactivate the computer, you must turn off the power once, and then turn it back on.
It will then be ready to receive new instructions.
Remember to check that the operating system is completely loaded first.

(4) The OSCHG command provides a set-up area.

2.10 Programmed Function Keys

2.10.1 Overview

The M20/23 Handy Business Computer has 14 programmed function keys (F1 to F7) by which the user can define up to 14 types of CLI command processing in combination with the SHIFT key.
They can be used as one-touch keys.
They are located along the top of the keyboard.

2.10.2 Using Programmed Function Keys

Once a programmed function key is defined, its contents are stored on the disk as part of the operating system and kept even after the power is turned off.
The user can assign different definitions to the same option key so that the definition of each option key on a particular disk can conventionally support the execution of programs stored on that disk.

> The M20/23 Handy Business Computer has 14 programmed function keys.
> Each programmed function key can be defined by up to 31 characters.
> When a programmed function key is pressed, its definition is entered in the computer in the same manner as are definitions entered from the keyboard.
> If a programmed function key remains undefined, no definition is entered in the computer even when the programmed function key is pressed.

2.10.3 Defining Programmed Function Keys

Each programmed function key can be defined as follows;

CLI command: FUNC ... defines a programmed function key.

FUNC [/S] [△filename] RETURN

S (switch) ... The current definition of the programmed function key is read into memory from the file following S, and is written on the disk.

Not specified

... The current definition of the programmed function key is read from the file into memory (when the file is not specified).

filename ..... indicates the file containing the definition of a programmed function key.
(Extension; .FNC)

FUNC RETURN

Command for programmed function keys

L ... displays the definition of a programmed function key.

E ... terminates the FUNC job.
The definition of programmed function keys is written into memory on a disk.

e ... terminates the FUNC job.
The definition of programmed function keys is not written into memory or onto disk.

s ... terminates the FUNC job.
The definition of programmed function keys is written into memory only.

R ... reads the definition of programmed function keys from the file into memory.

W ... writes the definition of programmed function keys onto the file.
You must enter the file name in reply to the displayed inquiry, Write file name ?.
File extension .FNC is automatically added to your specified file name.

F1 to F7
f1 to f7
..... identify 14 programmed function keys.

If F1 is pressed when F1 has been defined, its definition is deleted.

The definition of a programmed function key (F1 to F7 or f1 to f7) must be entered after the corresponding key is pressed.

For example, when F1 is pressed (F: alphabetic F1: numeric 1), the following is displayed:

COMMAND:F1 : (information entered from the keyboard) [RETURN]
[ESC]

Then, the information entered from the keyboard is assigned to programmed function key F1.

When [ESC] follows the information entered from the keyboard, another definition can be added to the current definition of the programmed function key.

The definitions of programmed function keys stored in the disk can be added to or changed.

```
READY
FUNC      [RETURN]
```

```
**************************************************************************
F1 : BASIC/D RUN'X <CR> :  (1) f1::
F2 : RUN'X  <CR>   :       (2) f2::
F3 : LIST 'SOUTA, 3<CR> :  (3) f3::
F4 : ABC :  (ESC)              f4::                    (4)
F5 ::                          f5::
F6 ::                          f6::
F7 ::                          f7::  BASIC/D RUN'
                                     OTHELO<CR>
**************************************************************************
```

```
COMMAND : F4  : ABC<CR>    (5)
COMMAND : E
READY
```

(NOTE) Δ ... indicates that the SPACE key is struck.
CR ... indicates that the RETURN key is struck.

(1) F1 is defined so that BASIC and program X can be loaded into memory and run when F1 is pressed.

(2) F2 is defined so that program X can be loaded and run when F2 is pressed.

(3) F3 is defined so that the program in memory can be printed when F3 is pressed.

(4) Programmed function keys f1 through f7 must be pressed as shown below:

[SMALL] + [Fn] (n=1 to 7)

(5) Numeric key [4] is pressed and ABC is assigned to F4.

(6) [E] indicates the end of the FUNC job.

(7) When [F1] is pressed, program X is executed immediately.

[L] displays the definition of programmed function keys.
When [RETURN] follows ABC, ABC is executed immediately.
When [ESC] follows ABC, the definition of the programmed function key is displayed.
Definitions can be added during execution.

2.10.4 Turn-Key Function

Programmed function key f7 has a kind of turn-key function.
When the following commands are assigned to programmed function key f7, they can be executed by pressing the programmed function key after bootstrapping.

(1) When a /CLI command followed by the [RETURN] key is assigned to the programmed function key, the CLI command is immediately executed after bootstrapping and setting the date.

(2) When a //CLI command followed by the [RETURN] key is assigned to the programmed function key, the CLI command is immediately executed after bootstrapping without setting the date.

(EXAMPLE)

```
//BASIC/D RUN'OTHELO      [RETURN]
```

2.11 Data Medium Protection

The M20/23 Handy Business Computer allows the user to protect the contents of floppy disks from being copied onto other floppy disks. This feature is called volume protection.

CLI command: VPROT ... protects a floppy disk volume or releases protection.

VPROT [/n/C/N/R/W] [RETURN]

/n ... Drive number subject to volume protection or protection release

/C ... Setting, deleting, and changing the password.

/N ... Releasing floppy disk volume protection in combination with /R or /W.

/R ... Setting read protection.

/W ... Setting write protection.

When VPROT is specified without switches, the current volume protection or password status is displayed.

(NOTE 1) Password need not be specified for volume protection.

(NOTE 2) When volume protection is released, a correct password must be specified (if a password has been assigned to the volume).

(NOTE 3) When a password is entered, M20/M23 does not display the password. The password must consist of up to alphanumeric characters except for [RETURN] and [CAN] keys.

(EXAMPLES)

Setting read protection

VPROT/R [RETURN]

Displaying volume protection status

VPROT [RETURN]

R PROTECTED ... indicates read protection.

NO PASS WORD ... indicates that no password has been assigned to the volume.

Releasing read protection

(Since no password has been assigned, the password entry can be omitted.)

VPROT/R/N [RETURN]

Setting write protection and passwords

VPROT/R/C

NO PASS WORD

NEW PASS WORD? ... Enter a password.
The password may consist of up to six alphanumeric characters.
Entry of the password must be followed by pressing the [RETURN] key.

Releasing write protection

(Since a password has been assigned, the password entry cannot be omitted.)

VPROT/W/N

PASS WORD? ... Enter the correct password.

WRONG PASS WORD ... This message indicates that an invalid password has been entered.

Changing the password

VPROT/C

OLD PASS WORD? ... Enter the current password.

NEW PASS WORD? ... Enter the new password.
If only the current password is to be deleted, the [RETURN] key must be pressed without entering the password.

2.12 Execution of BASIC Programs

A program will have one of the following two extensions is its name:

.SAV ... A program executable in CLI mode

.APU ... A program executable in BASIC mode -- only when BASIC is in main memory.

A program with the extension .SAV can be executed after loading in CLI mode.

If a program file named SALES.SAV is in the disk, you can execute SALES.SAV as illustrated below:

```
READY
SALES [RETURN]  ... SALES is in a disk as SALES.SAV.
  :
  :
  :
  :
    :
    : Execution of SALES.SAV
```

A program with the extension .APU can be executed after loading of BASIC.

BASIC can be loaded into main memory as illustrated below:

```
READY
BASIC [RETURN]
  :
* :
  :
  :
  :
  : indicates that M20/M23 enters BASIC mode.
  :
  :
  :
  :
```

This procedure can be omitted as illustrated below:

```
READY
BASIC/D  Δ RUN'BSALES Δ  [RETURN]
```

The program is called BSALES.APU.

This CLI command executes a program with extension .APU after BASIC and the program have been loaded into main memory.

CLI command: Loads BASIC and a program into main memory and executes the program.		
BASIC/D	RUN'[n:](program-name)	RETURN

(Example)

This example loads BASIC and program X (X.APU) into main memory, and executes the program.

```
READY
BASIC/D Δ RUN'X Δ RETURN
```

2.13 CLI Compound Commands

CLI commands can be grouped and specified as one command.
They can then be executed by pressing the RETURN key just once.

> CLI command: Executes a CLI command list
>
> (CLI command):(CLI command): ..... RETURN
>
> Don't forget to press the RETURN key after entering the last command.

> A compound command can contain up to 82 characters. If the RETURN key entry is counted, then the maximum characters allowable in a compound command is 83.

Several BASIC commands can also be combined and specified as one compound command.
For details, refer to a description of the GODBY statement in your BASIC manual.

2.14 OS Error Messages

When an error occurs during execution of the CLI commands or the part of a BASIC program, the operating system will display an error or warning message on the screen.

This kind of message is called an OS error message.

Message	Error/cause	Action/countermeasures
DISK FAIL	Error in reading from or writing onto the disk	1. Reset the disk in the drive. 2. Examine the disk for scratches.
ILLEGAL FILE NAME	An incorrect file name was specified.	Specify the file name correctly.
FILE DOES NOT EXIST	No file of the specified name is found.	Check the file name by means of the LIST command.
FILE ALREADY EXIST	The file name has already been registered into the directory.	Choose another name for the file.
FILE READ PROTECTED	An attempt was made to read a file with attribute R (Read prohibited).	Reexamine the file attribute and handle it correctly.
FILE WRITE PROTECTED	An attempt was made to write in a file with attribute W (Write prohibited).	Reexamine the file attribute and handle it correctly.
FILE IN USE	The file is in open status.	Close the file and call it again.

Message	Error/cause	Action/countermeasures
DISK SPACE EXHAUSTED	There is no sufficient area on the disk.	Rearrange the blocks on the disk with the SWEEP or DELETE command or use another disk.
FILE PROTECTED	An attempt was made to delete a file with attribute P (Permanent file).	Reexamine the file attribute and handle it correctly.
CHANNEL NOT IN USE	An attempt was made to write in or read a file for which no channel was open.	Open a channel for the file and continue processing.
CHANNEL ALREADY IN USE	An attempt was made to open a file which was already open.	This error occurs when the OPEN statement is used for the same channel more than once.
END OF FILE	An attempt was made to read or write although EOF appeared.	Check the number of records or the size of the file.
INVALID ACCESS MODE	An attempt was made to read or write into a file without regarding its file mode (continuous or sequential).	Reread the records on continuous and sequential files, and handle the file according to its attributes.
PARITY ERROR	A parity error has occurred.	Check the parity of the input data. See a description of the problems and countermeasures.

Message	Error/cause	Action/countermeasures
LINE BUFFER EXCEED	An attempt was made to input data exceeding the buffer capacity, that is, more than 83 characters.	Reexamine the length of the data, keeping the buffer capacity in mind, and reenter the data.
ILLEGAL COMMAND	An incorrect system call was made.	See a description of the system calls and correct the call.
ILLEGAL CLI COMMAND	A CLI command statement was entered incorrectly.	See a description of the CLI command and retry.

2.15 Disk Initialization Error

The operating system displays the following error message when a rattational speed error occurs during initialization of a floppy disk;

ROTATION SPEED ERROR mm nnnn

mm ..... indicates a track number (represented in hexadecimal) involving a rotational speed error.

nnnn ..... indicates a rotational speed (represented in hexadecimal).
If nnnn is smaller than $(1F4)_{16}$ the current rotational speed does not exceed the lower limit of the allowable rotational speed.
If nnnn is greater than $(1F4)_{16}$ the current rotational speed exceeds the upper limit of the allowable rotational speed.

The disk initialization routine (activated by the INIT command) normally operates when the current rotational speed of the floppy disk meets the following condition;

$$-8\% < (\text{Allowable Rotational Speed Error}) < +8\%$$

The rotational speed error can be calculated as follows:

$$\text{Rotational Speed Error } (\%) = \frac{(N-625)}{6250} \times 100.$$

where N is a decimal number equivalent to $(nnnn)_{16}$.

When a disk initialization error occurs, you should check the floppy disk and floppy disk drives as listed below:

(1) Check whether a rotational speed error occurs in a specific floppy

disk.

There is a possibility that a floppy disk may not normally rotate due to damage by careless handling.

When you utilize floppy disk media which are not supplied by SORD, SORD is not responsible for rotational speed error in the floppy disk media.

(2) Check the floppy disk drive when a rotational speed error occurs independently of floppy disk medium.

This check will be made by SORD customer engineers.

2.16 CLI Command Summary

CLI commands are entered in the operating system through the keyboard. They are issued when the computer is in ready status as illustrated below:

```
READY
(CLI command) ... Execution of the CLI command
READY
(CLI command) ... End of CLI command execution
      :
      :
      :
```

They are executed when the [RETURN] key is pressed.

1. Initializing a floppy disk

   INIT[/n] [RETURN]

   n ... indicates one of floppy disk drive #1, #2 or #3 associated with the disk to be initialized. You cannot specify floppy disk drive #0. The initialized disk has a file directory and will be used as a data storage medium.

2. Copying the disk

   COPY[/n] [RETURN]

   n ... indicates one of floppy disk 1, 2 or 3. Note that floppy disk drive #0 should not be used, and n should not be omitted.

   COPY $n_1 -> n_2$

3. Set-up

   SETUP[/D] [RETURN]

   D ... specifies that you start set-up from the value defined at shipment.

4. Dating

   DATE DD-MM-YY [RETURN]

5. Checking the used and unused areas on the disk

DISK[/n] RETURN

n ... specifies one of floppy disk drives #0 through #3.

n can be omitted only when floppy disk drive #0 is specified.

6. Listing the files on the disk

LIST[/n][/A][Δ SOUT/m] RETURN

n ... specifies one of floppy disk drives #0 through #3.

n can be omitted only when floppy disk drive #0 is specified.

[/A] ... Switch A is entered only when all file names (including OS component program names) are required.

If switch A is omitted, only the user's file names will be listed.

[Δ SOUT] ... The file names will be typed out on the printer.

If SOUT is omitted, the file names will be listed on the screen only.

m ... indicates a printer mode.

Switches W and S are also available to the LIST command.

7. Sweeping the disk medium

SWEEP[/n] RETURN

n ... specifies one of floppy disk drives #0 through #3.

n can be omitted only when floppy disk drive #0 is specified.

8. Creating a continuous file

CREATE Δ [n:](file-name)/C Δ n_1*m RETURN

n ... specifies one of floppy disk drives #0 through #3.

n can be omitted only when floppy disk drive #0

is specified.

file-name ...

specifies the name of a file to be created.

C ... specifies creation of a continuous file.

n_1*m ... n_1: number of records in the file

m : number of bytes in a record

9. Creating a sequential file

CREATE Δ [n:](file-name) RETURN

n ... specifies one of floppy disk drives #0 through #3.

n can be omitted only when floppy disk drive #0 is specified.

file-name ...

specifies the name of a file to be created.

10. Closing a file

CLOSE[/n] RETURN

n ... specifies one of floppy disk drives #0 through #3.

n can be omitted only when floppy disk drive #0 is specified.

11. Renaming a file

RENAME Δ [n_1:](file-name)Δ[n_2:](file-name)Δ RETURN

n ... specifies one of floppy disk drives #0 through #3.

n can be omitted only when floppy disk drive #0 is specified.

Note that the RENAME command renames a file in the same file directory.

file-name ...

specifies the name of the file to be renamed.

new-file-name ...

specifies the name to be newly assigned to the file.

12. Changing file characteristics

CHATR Δ [n:](file-name)[/A][/P][/R][/W][/X] RETURN

n ... specifies one of floppy disk drives #0 through #3.

n can be omitted only when floppy disk drive #0 is specified.

file-name ...

specifies the name of a file whose characteristics are to be changed.

13. Deleting a file

DELETE Δ [n:](file-name) RETURN

n ... specifies one of floppy disk drives #0 through #3.

n can be omitted only when floppy disk drive #0 is specified.

14. Transferring a file from one disk medium to another

XFER Δ [n_1:](file-name) Δ [n_2:](file-name) RETURN

n ... specifies one of floppy disk drives #0 through #3.

n can be omitted only when floppy disk drive #0 is specified.

A file associated with floppy disk drive n_1 is transferred a storage medium associated with floppy disk drive n_2 by the XFER command.

15. Replacing the operating system

OSCHG RETURN

The disk containing the new version of the operating system is set in floppy disk drive #0, and a disk containing the old version in floppy disk drive #1.

16. Programmed function keys

CLI command: FUNC ... defines a programmed function key.

FUNC[/S](file-name) RETURN

S (switch) ... The current definition of the programmed function key is read into main memory from the file following S and is written on the disk.
If S is omitted, the current definition of the programmed function key is read from the file into main memory (when the file is not specified).

file-name ... indicates the file containing the programmed function key definition.
(File extension: .FNC)

Commands for programmed function keys

L ... displays the definition of a programmed function key.

E ... terminates the FUNC job.
The definition of a programmed function key is written into main memory and onto a disk.

e ... terminates the FUNC job.
The definition of a programmed function key is written neither into main memory nor onto a disk.

s ... terminates the FUNC job.
The definition of programmed function key is written into main memory only.

R ... reads the definition of a programmed function key from the file into main memory.

W ... creates a file containing the definition of a programmed function key.
You must enter the file name in reply to the display inquiry, Write file name ?.
File extension .FNC is automatically added to your specified file name.

F1 to F7

f1 to f7

..... identify 14 programmed function keys.
If F1 is pressed when F1 has been defined, its definition is deleted.
The definition of a programmed function key (F1 to F7 or f1 to f7) must be entered after the corresponding key is pressed.
For example, when F1 is pressed (F: alphabetic F 1: numeric 1), the following is displayed:

COMMAND:F1 :(information entered from the keyboard) RETURN
ESC

Then, the information entered from the keyboard is assigned to programmed function key F1.
When ESC follows the information entered from the keyboard, another definition can be added to the current definition of the programmed function key.

17. Data medium protection

VPROT[/n][/C][/N][/R][/W] RETURN

n ... specifies one of floppy disk drives #0 through #3.
n can be omitted only when floppy disk drive #0 is specified.

/C ... Setting, deleting, and changing the password.

/N ... Releasing floppy disk volume protection in combination with /R or /W.

/R ... Setting read protection.

/W ... Setting write protection

When VPROT is specified without switches, the current volume protection or password status is displayed.

18. Execution of a BASIC program

READY

[n:]BASIC RETURN ... reads out the contents of the disk in floppy disk drive n.

* ..... indicates that M20/M23 enters BASIC mode.
M20/M23 waits for data entry through a keyboard.

(1) First, loads a BASIC program from a floppy disk into main memory.

(2) Second, execute it by pressing the RUN key.

[n:]BASIC/D Δ RUN'[n:](program-name Δ RETURN

19. CLI compound command

(CLI command) ; ..... RETURN

CLI commands are executed successively when the RETURN key is pressed.

One CLI compound command can consist of up to 83 characters including spaces.

Some CLI commands are not executable in compound commands.

If a compound command includes one of these, it will be ignored, and the system will return to ready status.